

Document Technique

Projet: création d'une page inscription/connexion



Chef de Projet: Quentin LAVAL
Responsable Technique: Bastien FONTAGNE
Développeur: Kevin LEMAIRE

CIEL2

I.Introduction

Ce document explique le fonctionnement et la structure de notre système d'inscription/connexion pour notre site web. Le site permet de s'inscrire, de se connecter et de se déconnecter.

Sommaire:

I.Introduction.....	2
2.Architecture.....	3
2.1.langage de programmation utilisé et technologie utilisé.....	3
2.2.Diagramme de l'architecture du site web.....	3
2.3.Diagramme de use case.....	4
2.4.Diagramme d'exigence.....	5
2.5.Diagramme de Gantt prévisionnel.....	5
3.Fonctionnalité.....	6
3.1.Inscription.....	6
3.2.Connexion.....	8
3.3.Déconnexion.....	10
4.Sécurité.....	11
5.Base De Donnée.....	11
5.1.Représentation visuel.....	11
6.Tests.....	12
6.1.Tests de l'inscription/connexion/déconnexion.....	12
6.2.Test de la bdd.....	12
7.Déploiement.....	13
8.Conclusion.....	13
9.Annexe.....	13
9.1Mail:.....	13
9.2Trello:.....	14
9.3Github:.....	14

2.Architecture

2.1.langage de programmation utilisé et technologie utilisé

Frontend: html/css et Javascript

Backend: Node.js

BDD: mysql/mariadb et phpmyadmin

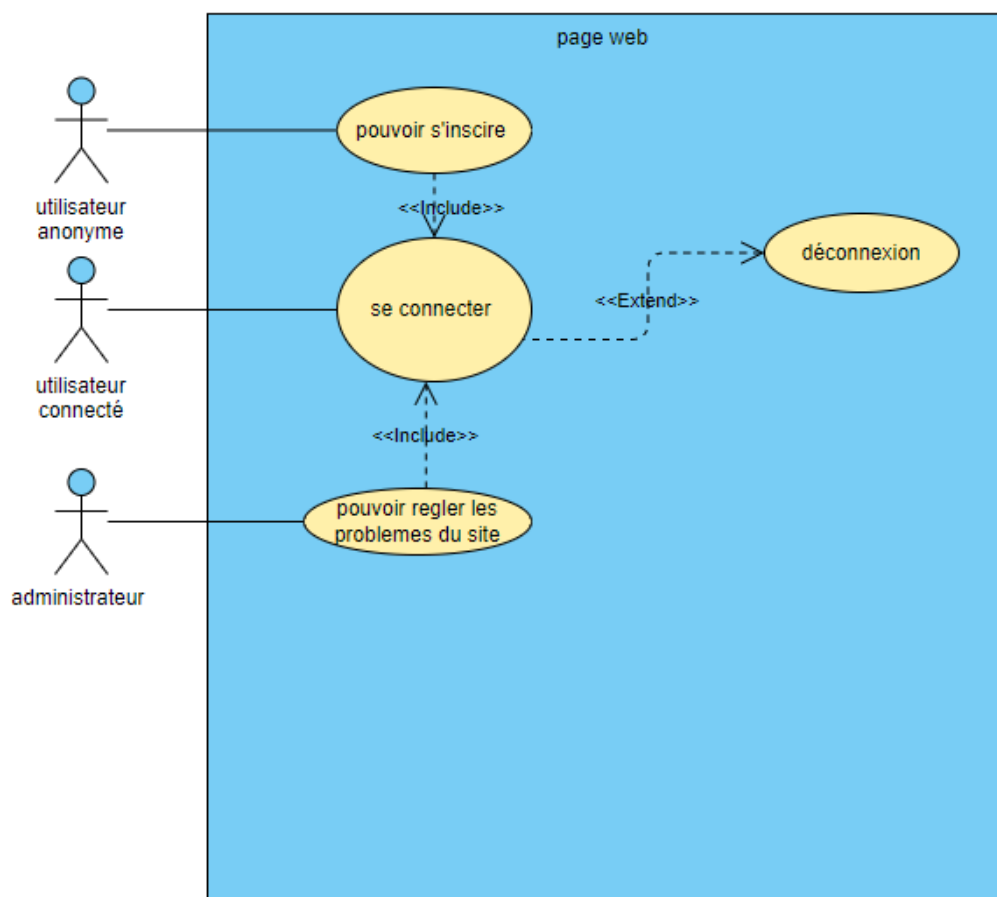
host du site web: sur la vm en n°219 (ip: 192.168.64.162) avec apache

Authentication: Token

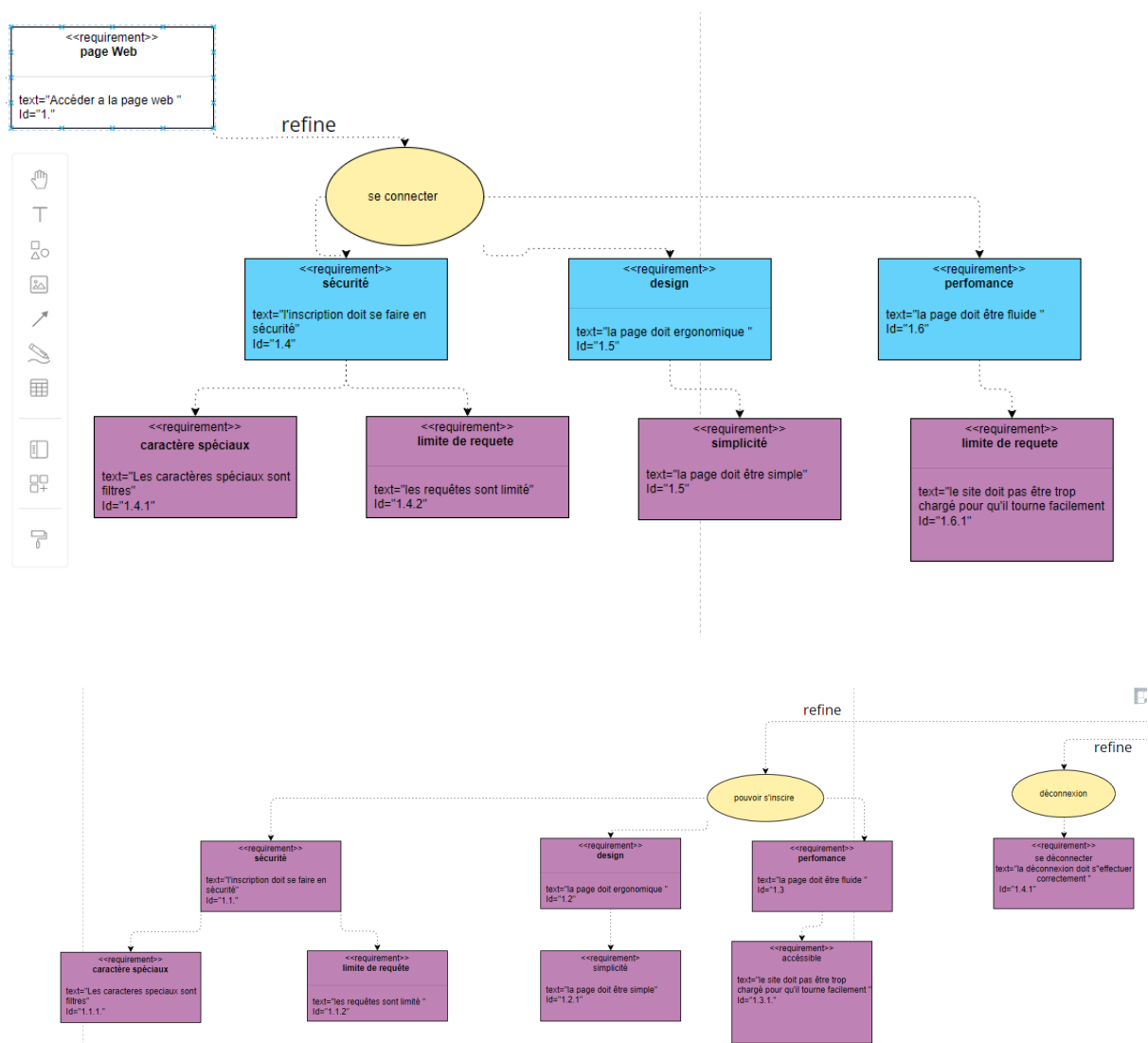
2.2.Diagramme de l'architecture du site web



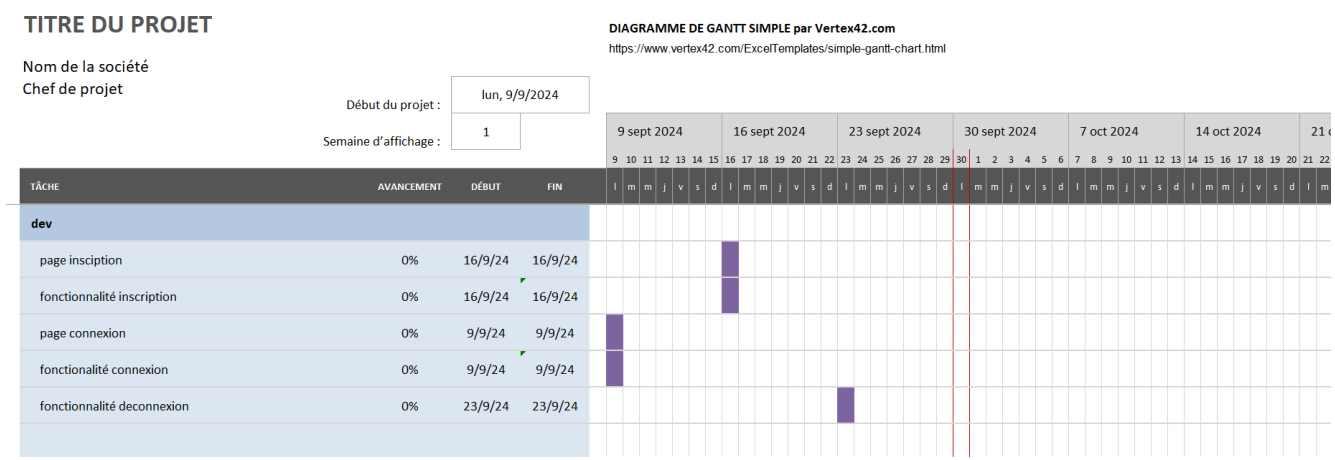
2.3. Diagramme de use case



2.4.Diagramme d'exigence



2.5.Diagramme de Gantt prévisionnel



Semaine d'affichage :				9 sept 2024							16 sept 2024							23 sept 2024							30 sept 2024						
				9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	3	4	5	6
TÂCHE	AVANCEMENT	DÉBUT	FIN	l	m	m	j	v	s	d	l	m	m	j	v	s	d	l	m	m	j	v	s	d	l	m	m	j	v	s	d
RT																															
mettre a jour vm	0%	9/9/24	9/9/24																												
mettre a jour , changer mdp vm , phpmyadmin	0%	9/9/24	9/9/24																												
créer bdd	0%	9/9/24	9/9/24																												
créer github	0%	9/9/24	9/9/24																												
MCD	0%	16/9/24	16/9/24																												
doc tehcnique	0%	16/9/24	16/9/24																												

Semaine d'affichage :				9 sept 2024							16 sept 2024							23 sept 2024							30 sept 2024						
				9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	3	4	5	6
TÂCHE	AVANCEMENT	DÉBUT	FIN	l	m	m	j	v	s	d	l	m	m	j	v	s	d	l	m	m	j	v	s	d	l	m	m	j	v	s	d
CP																															
grantt		9/9/24	9/9/24																												
use case		9/9/24	9/9/24																												
diagramm d exigence		9/9/24	16/9/24																												
cahier des recettes		16/9/24	23/9/24																												
creation du word		23/9/24	23/9/24																												

3.Fonctionnalité

3.1.Inscription

Route: /register

Méthode: POST

Code: //Route d'inscription via l'API

```
app.post('/register', (req, res) => {

    const {mail, name, password} = req.body;
    if(!mail || !name || !password) return res.status(400).json({
message : "Le nom d'utilisateur ou le mot de passe est manquant !" });

    const sqlVerif = `SELECT mail FROM User WHERE mail = ?`

    connection.query(sqlVerif, [mail], (err, result) => {
        if(err) {
```

```

        console.error("Erreur lors de la vérification
d'inscription\nErreur SQL :\n",err);
        return res.status(500).json({ message: "Erreur lors de la
création du compte" });
    }

    if(result.length === 0) {
        bcrypt.genSalt(10, (err, salt) => {
            if(err) {
                console.error("Erreur lors de la génération du sel
Bcrypt\nErreur Bcrypt Salt:\n",err);
                return res.status(500).json({ message: "Erreur lors
de la création du compte" });
            }

            bcrypt.hash(password, salt, (hashErr, hashedPassword)
=> {
                if (hashErr) {
                    console.error('Erreur lors du hachage du mot de
passe\nErreur Bcrypt Hash :\n', hashErr);
                    return res.status(500).json({ message: "Erreur
lors de la création du compte" });
                }

                let token = jwt.sign({ mail }, jwtKey);

                const sqlNewUser = `INSERT INTO User (mail, name,
password, token) VALUES (?, ?, ?)`;

                connection.query(sqlNewUser, [mail, hashedPassword,
token], (sqlErr) => {
                    if(sqlErr) {
                        console.error("Erreur lors de la création
du nouvel User\nErreur sql :\n", sqlErr);
                        return res.status(500).json({ message :
"Erreur lors de la création du compte" });
                    }
                    return res.status(200).json(
                        {
                            message: "Compte créé avec succès",
                            token: token
                        }
                    );
                });
            }
        });
    }
}

```

```

        })
    })
    })
    } else {
        return res.status(409).json({ message : "Nom déjà utilisé"
    })
    }
    })
    });

```

Traitement:

- vérifie si le mail ou le nom ou le mot de passe sont manquant
- vérifie si il y a eu des erreurs lors de l'inscription
- vérifie si le mot de passe a été haché et si il y a eu une erreur dans sa génération
- vérifie si il y a eu une erreur dans la création du compte
- vérifie si le nom a déjà été utilisé
- créer le compte

Réponse:

200: "Compte créé avec succès"

400: "Le nom d'utilisateur ou le mot de passe sont manquants"

409: "Nom déjà utilisé"

500: "Erreur lors de la création du compte"

3.2.Connexion

route: /login

méthode: POST

code: //Route de connexion via l'API

```

app.post('/login', (req, res) => {
    const { mail, password } = req.body;

    if(!mail || !password) return res.status(400).json({ message: "Nom
ou mot de passe manquant !" });

    const sqlLogin = 'SELECT password, token FROM User WHERE mail = ?'

    connection.query(sqlLogin, [mail], (err, result) => {

```



```

        if(err) {
            console.error("Erreur lors de la requête SQL login\nErreur
sql :\n",err);
            return res.status(500).json({ message: "Erreur lors de la
connexion" });
        }

        if(result.length === 0) {
            return res.status(409).json({ message: "Nom inconnu" });
        }

        const isPasswordValid = bcrypt.compareSync(password,
result[0].password);

        if(isPasswordValid) {
            const sqlLoginToken = 'UPDATE User SET token = ? mail = ?'
            let token = jwt.sign({ mail }, jwtKey);

            connection.query(sqlLoginToken, [token, mail], (err) => {
                if(err) {
                    console.error("Erreur lors de la requête
LoginToken\nErreur sql :\n", err);
                    return res.status(500).json({ message : " Erreur
lors de la création du token"});
                }

                return res.status(200).json(
                    {
                        message : "Connexion Réussie",
                        token : token
                    }
                )
            })
        } else {
            res.status(409).json({ message : "Mot de passe
incorrect"});
        }
    })
})

```

Traitement:

- vérifie si le mail, le nom ou le mot de passe sont manquants
- vérifie si il y a une erreur durant la requête sql
- vérifie si le nom est dans la bdd
- vérifie le token de connexion

Réponse:

200: "Connexion Réussie"

400: "Nom d'utilisateur ou mot de passe manquant"

409: "Nom inconnu" ou "Mot de passe incorrect"

500: "Erreur lors de la connexion" ou "Erreur lors de la création du token"

3.3.Déconnexion

route: /disconnect

méthode: POST

code: //Route de déconnexion via l'API

```
app.post('/disconnect', (req, res) => {
  const {token} = req.body;

  if(!token) return res.status(400).json({ message: "Le token du User est requis pour la déconnexion"});

  const sqlDisconnect = 'UPDATE User SET token = NULL WHERE token = ?'

  connection.query(sqlDisconnect, [token], (err) => {
    if(err) {
      console.error("Erreur lors de la requête de déconnexion\nErreur sql:\n", err);
      return res.status(500).json({ message : " Erreur lors de la déconnexion"});
    }
    return res.status(200).json({ message: " Utilisateur déconnecté"});
  })
})
```

Traitement:

- vérifie si il y a le token de l'utilisateur requis pour la déconnexion
- vérifie la requête de déconnexion

Réponse:

200: "Utilisateur déconnecté"

400: "Le token du User est requis pour la déconnexion"

500: "Erreur lors de la déconnexion"

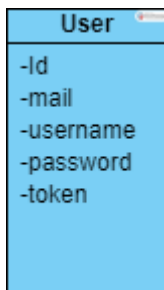
4.Sécurité

cryptage du mot de passe: les mots de passe sont hachés en utilisant bcrypte

token: les token sont créer en utilisant JWT, ils ont une durée limité et ont une clé secrète

5.Base De Donnée

5.1.Représentation visuel



6.Tests

6.1.Tests de l'inscription/connexion/déconnexion

Description	Critère d'acceptation	Résultat
-------------	-----------------------	----------

1.Créer un compte	Le compte est créer avec un nom d'utilisateur, un e-mail et un mot de passe	ok
2.Se connecter au compte	On se connecte au compte en utilisant le nom d'utilisateur, le mail et le mot de passe associé	ok
3.Se déconnecter du compte	On se déconnecte du compte	ok

6.2.Test de la bdd

Description	Critère d'acceptation	Résultat
1.Création et stockage des donnès	On vérifie si les information de connexion sont bien stocker et crypter dans la bdd	ok
2.Création du token	On vérifie si le compte est bien liée à un token	ok

7.Déploiement

On déploie le site sur la vm n°219 (192.168.64.162)fr pour tester l'application

8.Conclusion

Ce document technique montre comment implémenter et sécuriser le système d’inscription et de connexion pour le site web, ce dernier sera enrichi au fur et à mesure des projets.

9.Annexe

9.1Mail:

1 sur 2

<

>

Ft

compte rendu le 16/09/2024

Boîte de réception x

q

quentin laval

<laval.quentin80@gmail.com>

À julienlanglace, lemaire.kevin6020, fontagne.bastien1

Bonjours a tous voici le compte rendu de fin de journée

Nous avons prévu pour cette séance les choses suivantes

Kevin : pages de connections fonctionnelles

Bastien : faire le mcd et commencer le cahier de recette

Quentin faire le use case et commencer le diagramme d'exigence

Pour ma part je suis a l heure

Pour la semaine prochaine sa sera le dernier cours pour finaliser le projet

Kevin: finaliser le site

Quentin et Bastien : faire le diapo

lun. 16 sept. 17:27

☆

😊

↩

⋮

f

Bastien Fontagne

À moi

Tout est ok pour moi

lun. 16 sept. 17:29

☆

😊

↩

⋮

l

Kevin Lemaire

À moi

tout est bon pour moi

lun. 16 sept. 17:32

☆

😊

↩

⋮

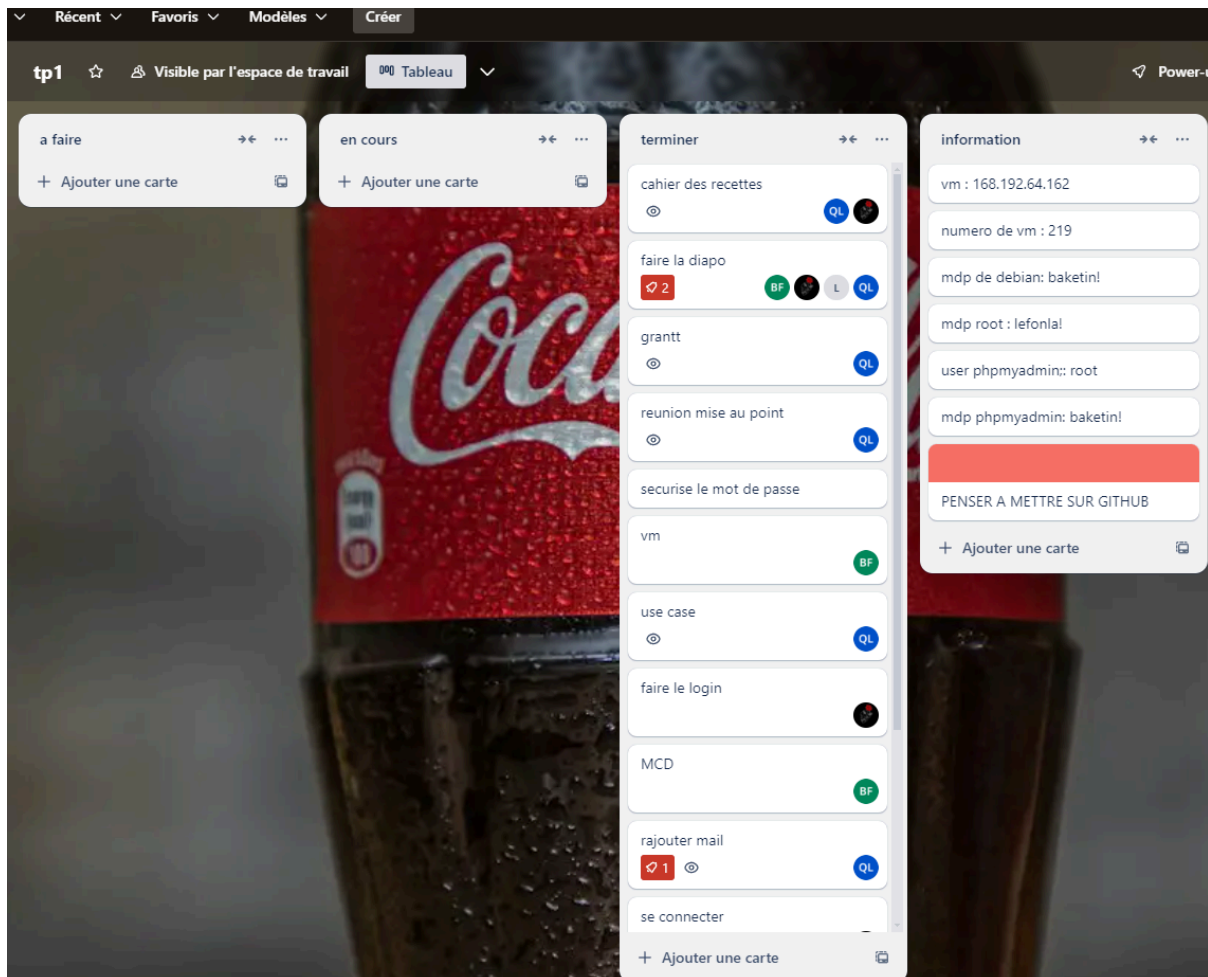
↩ Répondre

➡ Transférer

😊

🧠 IA Réponse

9.2Trello:



9.3Github:

